

ESD-TR-67-371
ESTI FILE COPY

ESD RECORD COPY

RETURN TO
SCIENTIFIC & TECHNICAL INFORMATION DIVISION
(ESTI) BUILDING 121A

ESVP

MERGE

1 2
ESD ACCESSION LIST
ESTI Call No. AL 57898
Copy No. 1 of 2 cys

AUGUST 1967

I. Beilin

Prepared for
DEPUTY FOR COMMAND SYSTEMS
COMPUTER AND DISPLAY DIVISION
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts



This document has been approved for public release and sale; its distribution is unlimited.

Project 503F

Prepared by

THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-5165

AD0658446

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

MERGE

AUGUST 1967

I. Beilin

Prepared for
DEPUTY FOR COMMAND SYSTEMS
COMPUTER AND DISPLAY DIVISION
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts



This document has been approved for public release and sale; its distribution is unlimited.

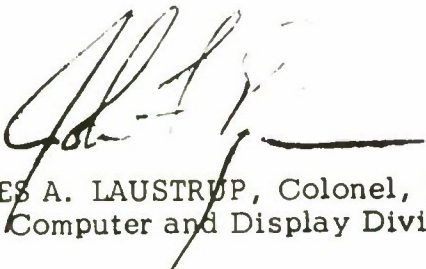
Project 503F
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-5165

FOREWORD

This document was prepared by The MITRE Corporation for the Deputy for Command Systems, Computer and Display Division, of the Electronic Systems Division, Air Force Systems Command, L. G. Hanscom Field, Bedford, Massachusetts, under Contract AF 19(628)-5165.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved.


for CHARLES A. LAUSTROP, Colonel, USAF
Chief, Computer and Display Division

ABSTRACT

A general purpose 7030 computer program has been written in the FORTRAN environment to merge, in a single run of the program, two, three, or four tape files which are the resulting output from a sort program. These files may be more than one tape in length and may be either FORTRAN or non-FORTRAN formatted. Input and output processors have been designed to handle various types of item formats; the program accepts items of fixed length, variable length (identified by a predesigned terminator) and variable length with a length-defining field. Upon request, the output processor will eliminate duplicate items and/or fill in variable length records to a standard size.

TABLE OF CONTENTS

		<u>Page</u>
SECTION I	INTRODUCTION	1
SECTION II	PROGRAM FUNCTIONS	3
	Input Channel Tape (ICT) Tables	3
	Input/Output Queue Tables	5
	Input Buffer Queue Tables	7
	Empty Input Buffer Stack (EIS and NEIS)	
	Word and Table	10
	Output Buffer Queue (OP1) Table	11
	Empty Output Buffer Stack (EOS and NEOS)	
	Word and Table	12
	IOD Table	13
	File Unit Table (FUTf)	13
	BLIM1 and BLIM5 Tables	14
	Input Edit	15
	Output Edit	16
SECTION III	SUBROUTINES	17
	List of Subroutines	19
SECTION IV	COMMON STORAGE	20
	Tables	20
	Words In Common Storage	21
SECTION V	CONTROL DECK ARRANGEMENT	24
	MERGEP1 Card	24
	MERGEP2 Cards	26
	MERGEP3 Cards	28
	MERGEP4 Cards	28
APPENDIX	FOUR-WAY COMPARISON AND RETURN TREE	33

SECTION I

INTRODUCTION

A general purpose 7030 computer program, MERGE, has been written in the FORTRAN environment to merge in a single run of the program two, three, or four tape files which are the resulting output from a sort program. These files may be more than one tape in length and may be either FORTRAN or non-FORTRAN formatted. Input and output processors have been designed to handle various types of item formats; the program accepts items of fixed length, variable length (identified by a predesignated terminator) and variable length with a length-defining field. Upon request, the output processor will eliminate duplicate items and/or fill in variable length records to a standard size.

The major functions of the program are illustrated in Figure 1. The flow as shown is only an introduction to the general functions of the program; the actual flow varies according to the intermixing of

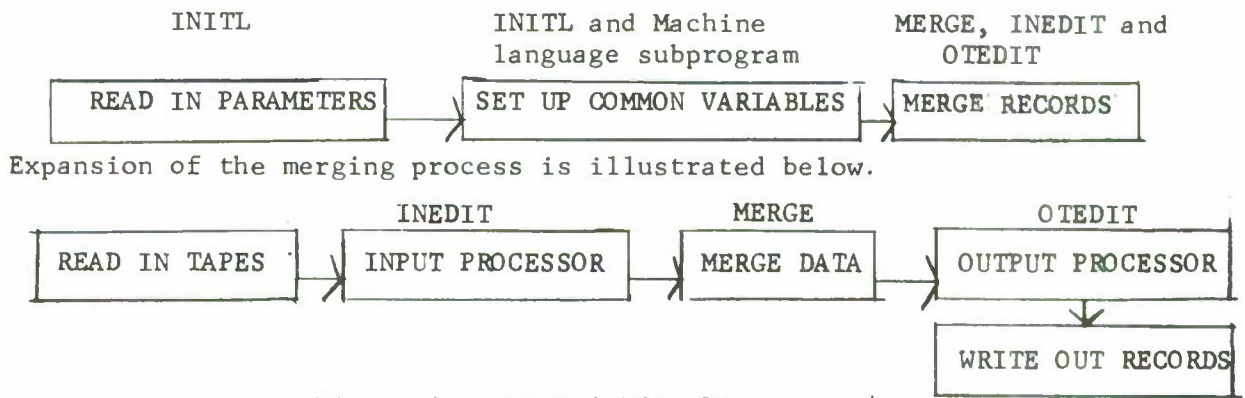


Figure 1. General Flow Diagram

mainstream and autostacked modes. The relative positions of the programs is of major importance since some of the programs are eventually destroyed; therefore, a knowledge of core layout is essential. A description of the subroutines - and the order in which they should be arranged - is provided in the section entitled SUBROUTINES.

SECTION II
PROGRAM FUNCTIONS

The first function of the program MERGE is to find data for each of the (four) files. One input is preselected from each channel before the program proceeds. This process requires that tables be structured to keep up with the correct address and to modify the appropriate address of the last file for each channel (LSTA-D). This section describes in detail the basic table structures in the program and the conditions under which they are modified.

Input Channel Tape (ICT) Tables

The table word format is described in the paragraph on common storage tables; the indicators included in this table and the functions they perform are described below.

The channel BUSY bit is turned on only if there is a read or write being performed on the channel. The contents of the control word reflect the I/O operation currently on the channel; bit 27 of the control word is 1 if this is a write operation; a 0 if it is a read operation.

The address for the last file of each item is located in LSTA-D (words 4, 5, 6, and 7) which point to "HIKEY" - which contains a floating point mantissa of ones - whenever:

input does not exist, e.g., input D during a two or three way
merge is blank
the specific input file has been depleted
the tape containing this input file is connected to another
I/O channel
the channel is already reading that file.

Since HIKEY is greater than any other key, the input is not selected.
The INITL subroutine clears the tables; computes the number of input
files not on this channel, called OTHER, in bits 32-49 of word 0;
finds the channel I/O queue address (QP); and points to HIKEY.

Conditions Under Which ICT Tables Are Modified

The BUSY bit is set to 1 when a read or write operation is begun and
to 0 before exiting from the read or write EOP, unless the next read
or write is begun there. OTHER is adjusted whenever a tape swap in-
volves a channel change.

The control word (word 1) must be filled with the word fetched from
the top of the channel I/O queue before a read or write is begun; this
word becomes the read or write control word.

The QP word should always point to the top of the channel's I/O queue;
its count field (word 3, bits 28 - 45) must contain the number of entries
currently on that queue.

After an input is selected for reading, its LST must be pointed to HIKEY. At read EOP time, the LST must point to the core address of the last item in the buffer just filled. This action permits a procedure called "preselection" to pick the next read to queue by deciding which of the inputs on that channel will be depleted first.

Input/Output Queue Tables

There are four queues, one for each data channel assignment now possible and one for future expansion. The I/O queues are in a one-to-one correspondence with the I/O channel tables. The I/O queue table is accessed via the I/O queue pointer, QP, of the corresponding I/O channel table. The count field of QP contains the number of entries currently in the queue (word 3, bits 24 - 45 of ICT tables).

The subroutine INITL computes the necessary size of the I/O queue tables, which is the total number of input buffers allocated for the merge plus the total number of output buffers allocated for the merge plus 1. The format varies depending upon the content; if the queue is empty, all words are zero, if the queue is not empty the entries have two forms: read requests and write requests.

Read Requests XW, IB, L, FUT_f

where XW = index word

IB = input buffer

L = length

FUT_f = input file pointers, $2 \leq f \leq 5$

bit 27 = \emptyset

When the next input is selected, either at read EOP time or at input buffer depletion time if busy = \emptyset , a read request in the form shown above should be constructed as follows and put at the end of the queue:

1. a buffer address (IB) is obtained from the Empty Input Buffer Stack (EIS) and inserted into the value field.
2. the count field is set to the standard input buffer length (L)
3. the refill field is set to the address of the selected input File Unit Table (FUT_f; $2 \leq f \leq 5$).

Write Requests XW, OB, M, FUT_{1,1}

where XW = Index word

OB = Output buffer

FUT₁ = File Unit Table 1

bit 27 = 1

When an output buffer is filled, a write control word, includes a 1 in bit 27, is constructed and put on the end of the output I/O queue. This action constitutes a request to write the buffer contents. The value field (OB) of the control word specifies the core origin of the buffer; the count field (M) specifies the record length. The refill field must point to the output File Unit Table (FUT₁) entry. When available, a new current entry buffer must be put on the top of the output buffer queue.

Conditions Under Which I/O Queue Tables Are Modified

When the current read or write operation on the channel has been completed and its control word disposed of, the top word - if any - in the I/O queue is moved to the control word (word 1) of the same I/O

channel table (ICT) and the queue is "popped". The control word just fetched is used to begin execution of the next read or write on the channel.

Each new request should be made from the top of the queue. The contents of the Input Buffer Queue - including the terminator word - should be moved up one word (popped) and the QP count decremented.

Input Buffer Queue Tables

There are four input buffer queue tables, one for each logical input file. Each queue contains a control word for each buffer currently available in core. These buffers are accessed via QP2, QP3, QP4, and QP5 - which point respectively to the A, B, C, and D input buffer queues. Each input queue pointer has the form:

QP_f: XW, BQ_f, N_f

where QP_f = QP2, QP3, QP4, or QP5

BQ_f = pointer to the queue origin and f = 2, 3, 4, or 5

N_f = the current number of entries and f is same as above

Allocation of storage and the presetting of QP2, QP3, QP4, and QP5 is performed by the subroutine INITL. Each word in the Input Buffer Queue table has either the form:

XW, IB_i, L, FUT_f

where XW = index word

IB_i = input buffer

L = length

FUT_f = File Unit Tape; $2 \leq f \leq 5$

or the form: a terminator word of zero following the last entry.

The top control word in each input buffer queue points to the top of the current input buffer. Merging is accomplished by examining the current item of each input buffer and selecting the one with the least key, moving it (with output editing) to the "current" output buffer, stepping the output buffer index register and the buffer index of the selected input. \$A, \$B, \$C, \$D, and \$O are symbolic names for the four input index registers and the output index register.

Conditions Under Which The Input Buffer Queue Table is Modified

When an input buffer is deleted, its address is returned to the Empty Input Buffer Stack (EIS) pointer and the input buffer queue "popped". If this EIS word is \emptyset , no buffer containing this input is available. The program design allows this condition to occur infrequently. If the input channel BUSY bit = \emptyset , the next read on this channel should be preselected, a read request put on the channel I/O is in progress. Preselection and setup of the next read should occur normally when the operation is completed. The merge should loop until the queue is eventually $\neq \emptyset$ - which indicates that a buffer is available. If the word at the top of the queue $\neq \emptyset$, it points to a new current buffer for this input. If the BUSY bit = \emptyset , a read request is preselected, stacked, and the top I/O queue word is unstacked and executed. The input index register should be reset to point to the new queue and merging resumed.

Whenever a read is completed, LST in the appropriate I/O channel table should be set to the address of the last item in the record which is used in preselecting the next input to be read on this channel. The completed read control word (in the I/O channel table word) is put on the end of the input buffer queue and the queue-pointer count field is incremented. The next input to be read from the channel is preselected, a read request constructed and added to the end of the read queue and, if BUSY bit = \emptyset , the top request "popped" and initiated. When an input is exhausted, a dummy control word, HIKEY, which points to a key of bits, is put on the end of its input buffer queue. This control word will eventually rise to the top of the queue and prevent that input from being selected and merged. Any other read request for this input is deleted from the I/O queue; the buffer address is returned to the Empty Input Buffer Stack (EIS). Preselection proceeds as described above.

When an intermediary input reel - but not a file - is exhausted, the current reel should be unloaded and the current and alternate IOD's should be switched (the IOD table is also described in this section). Then, if the current and alternate IOD's are assigned to the same channel, the read is simply repeated using the same control word (word 1 of ICT) and the new IOD.

When the end of an intermediary input reel involves a channel address change, because the new current IOD has a different channel address from the previous IOD, considerable adjustment is required to purge the old queue, and to adjust the new one.

The following operations are performed when in the Autostack mode.

1. Decrement the old OTHER; increment the new OTHER.
2. Delete all read requests from the old input I/O queue, if any, for this input only - returning the released buffers to the EIS.
3. Delete all read requests (for all inputs) from the new input I/O queue, returning the released buffers to the EIS; retain any write requests on this queue.
4. Examine input buffer queues of all inputs now on the new channel. A read request is put on this channel I/O queue for any such input whose "current" buffer is empty.
5. If there is no empty "current" buffer, set LST for the newly transferred item position in the last of its standby buffers. The preselection process is then begun.
6. Unstack and begin execution of the next I/O request on the old channel.

Empty Input Buffer Stack (EIS and NEIS) Word And Table

The NEIS table contains the available buffers that are to be used for input I/O. Access to this table is via the word EIS which has the following form:

XW, NEIS, M

where XW = index word

NEIS = location of the table

M = the current number of empty input buffers; $0 \leq M \leq$ maximum number of input buffers

The Subroutine INITL computes the amount of available core and then assigns the starting addresses of each input buffer. These results are placed in the NEIS table and in the one word EIS.

Conditions Under Which The Empty Input Buffer Stack Table is Modified

An entry is withdrawn and the stack 'popped' each time a read is queued and an entry is returned to the NEIS table; and the stack is 'pushed' each time any input buffer is emptied or a queued read is abandoned - e.g., when a queue is merged or at the end of input. A read is not initiated unless the stack contains at least one entry. Unless a channel I/O queue is empty, the program may not queue another read for it unless the number of available buffers is greater than OTHER for this channel. This restriction is intended to prevent a channel from cleaning the stack and forcing delays on other channels due to lack of buffers.

Output Buffer Queue (QP1) Table

This queue points to a word which points to the buffer currently being filled by the merge process. QP1 has the form:

QP1: XW, BQ1, 1

where XW = index word

BQ1 = pointer to the output buffer location currently being filled BQ1 has the form:

XW, OB_j, M_j

where OB_j = the location of the output buffer

M_j = the number of words presently in the table

Conditions Under Which the Output Buffer Queue Table is Modified

When the output buffer OB_j is filled, a control word for this buffer is constructed and put in the proper I/O queue table. A new output buffer is fetched from the NEOS and placed in the value field of BQ_1 ; the count field of BQ_1 is set to \emptyset .

Empty Output Buffer Stack (EOS and NEOS) Word and Table

The second function of the MERGE program is to fetch an output buffer from the Empty Output Buffer Stack (EOS), construct a zero field and put it in the top queue position to be ready to accept the first merged item.

The NEOS table contains the starting addresses of the available output buffers. They are accessed by the Empty Output Stack (EOS) pointer, which has the form:

$XW, NEOS, N$

where XW = index register

$NEOS$ = starting address of the table

N = the current number of empty output buffers; $\emptyset \leq N \leq$ maximum number of buffers allocated for the MERGE by INIIL.

Conditions Under Which The Empty Output Buffer Stack Is Modified

Each time the current output buffer is filled, an entry is transferred into the current output buffer and the stack is 'popped'. If NEOS is empty, MERGE loops until a completed write EOP restores at least one entry to NEOS. Whenever a write is completed, the released buffer address should be returned to the EOS and the stack "pushed".

IOD Table

This sixteen-register associates an IOD with an I/O channel table, which is directly accessed. Each entry has the symbolic base address IODd where d corresponds to a possible merge tape IOD(MCP reference number). It may also be accessed via File Unit Table entry (FUTf; q.v.).

The format of IOD is:

IODd: VF, d; VF, \emptyset (if no merge tape has IOD assignment d)

or IOD : VF, d; VF, ICTK

where VF = value field

d = IOD number $\emptyset \leq d \leq 15$

ICTK = an index address corresponding to the 7030 tape channel to which the tape drive is connected; where, normally, $1 < K < 4$; ICT is the I/O channel table.

The subroutine INITL looks up each merge tape channel assignment in the IOD's MCP unit area table and enters the base address of the channel I/O Control Table (ICT) in the right half of the IOD table entry. These assignments do not change.

File Unit Table (FUTf)

The File Unit Table associates each logical input and output with its current and alternate tape drive and indirectly with the I/O channel table. Each entry of the table is directly addressable. By convention, f = 1 corresponds to the output file; f = 2 corresponds to the first input file (INPUT A), f = 3 corresponds to the second input file, etc. The FUT may be accessed also by the refill field of any control word.

The format is as follows:

FUTf: VF, IODd; VF, IODd'

where IODd = the IOD table entry for the logical files current tape drive

IODd' = the entry for the logical file's alternate tape drive, if any.

If only one tape drive is assigned, then IODd and IODd' are equal.

Whenever an EE occurs the half-words are switched: The final EE points to IODØ which always has the form:

IODØ: VF, Ø; VF, Ø

indicating that no tape unit is attached for that file.

BLIM1 and BLIM5 Tables

These tables perform the same function for the current buffer limiting address that FUT1 - FUT5 perform for the input/output files. This value enables the program to identify the last item in a buffer; no initialization is required. Modification of the table occurs each time a new buffer is added.

Input Edit

The edit program preprocesses the tape record into item blocks whose format is compatible with MERGE. The keys of each item are extracted and, if requested, are converted into a modifiable collated key. These keys are stored in the 48-bit mantissas of the words preceding the item. The last 24 bits of the mantissa of the appended words contain an item number which is always one more than the previous one. For example, a record containing 14 items of 210 characters (6-bit code) each would be processed as illustrated in Figure 2. Assume 2 keys in characters 5-7, 10-19.

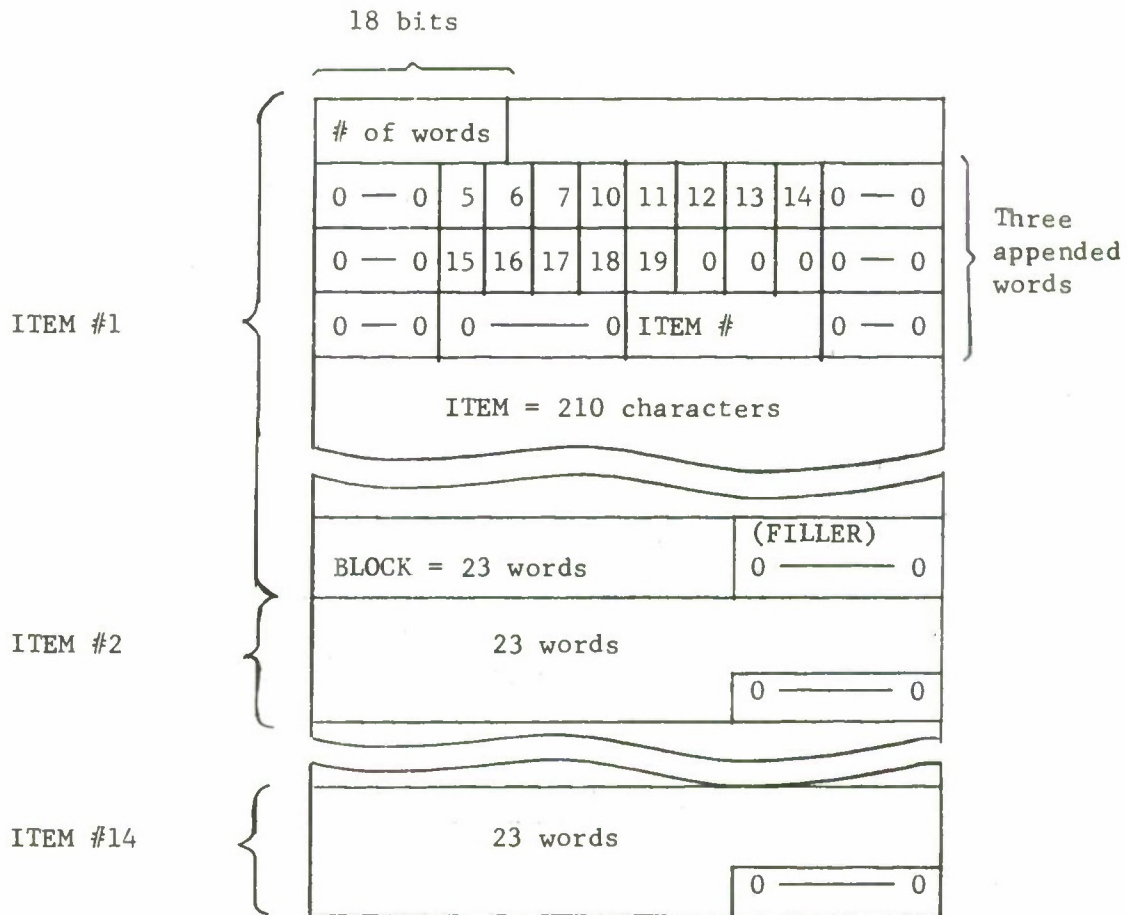


Figure 2. Format of Appended Key Words

Since the 48-bit mantissa of the second appended word is not large enough to contain the key characters and the 24-bit item number, the item number is right justified in the mantissa of the third appended word. The pointer is moved 23 words each time a new item is required, the 23rd word of each block contains a 20-bit filler of zeros. The preprocessor is capable of handling general input tapes. The description of the tapes and the preprocessing desired is specified in the data control cards, described in the section entitled CONTROL DECK ARRANGEMENT. The collating table is stored in common storage and may be modified by a block data deck, described in the FORTRAN Manual. The standard collating sequence is:

blank, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, =, ' , +, A, B, C, D, E,
F, G, H, I, . ,) , - , J, K, L, M, N, O, P, Q, R, \$, *, /, S,
T, U, V, W, X, Y, Z, ≠, , , (.

During initialization the program sets up the code to preprocess the input tapes.

Output Edit

The program takes the item selected by the merge process, sequence checks it for error, strips the appended key words (in the example in Figure 2, the first three words), and stores the item in the buffer. The program will also, upon request, eliminate duplicates. Output tape and editing are specified as desired through the data control deck. During initialization these parameters are used to generate the code to be used for input processing.

SECTION III

SUBROUTINES

- MAIN - This subroutine is coded in FORTRAN; its purpose is to call INITL and MERGE routines.
- INEDIT - This subroutine is called when the program is in autostack mode at EOP time of a read input tape request. In this phase, the preprocessor extracts the sort keys, converts the data according to a collating sequence - if this option is requested, and rearranges the data into a form compatible with the MERGE operation.
- OTEDIT - The output edit is called by the mainstream program when placing the item in the output buffers. The output edit strips the sort keys from the item and packs the item into the buffer. Upon request, the item will be eliminated if it is a duplicate of a previous one.
- MERGE - This program performs all of the merging and input/output operations; INEDIT and OTEDIT are called by MERGE when needed. The following words and routines are all entries incorporated into one so-called subprogram which must be placed after MERGE; INITL calls these entries as needed.
- FUTF - adds IODØ to every half-word in the FUT tables.
- IODF - Locates the proper channel and stores ICT pointers in the right half-word for each unit in the IOD table that is being used.
- ZEROA - Initializes tables, clears QP tables, FUT tables, and ICT tables and stores zeros in each.

FIXDT - Arranges common storage words in the proper order for general parameters.

INFIX - Arranges common storage words in the proper order for input parameters.

OUTFIX - Arranges common storage words in the proper order for output parameters.

SEOS - sets up pointers for EIS and NEIS.

FXAF - sets up LSTA - D words to point to HIKEY.

FBUF - computes length of available buffer and table space.

WDCON - converts A8 code formats into proper MERGE format; used for filler and terminator bits.

OTHER - contains the number of input files not presently on a specific channel.

INITL - This program extracts the parameters, and initializes the common storage variables. When the initialization phase is over, the core storage areas occupied by the programs FUTF to INITL are used as buffer space. Thus, all space in core following the MERGE program and preceding common storage, is occupied by tables.

IBIN, IXTC, and KOMPl - These routines are described in the 7030 Facility Manual.

List of Subroutines

MERGE subroutines are entered in the following order. Since some of the routines are cleared out of core after being used, the order is of utmost importance.

MAIN
INEDIT
OTEDIT
MERGE
FUTF
IODF
OTHER
ZEROA
FIXDT
SEOS
FXAF
FBUF
WDCON
INFIX
OUTFIX
INITL
IBIN
IXTC
KOMP1

SECTION IV

COMMON STORAGE

Tables

The following is a description of the tables to be found in common storage.

IOD0 - references the input/output device

FUT1 - FUT5 - associate logical input/output with current and alternate tape drives

BLIM1 - BLIM5 - identifies address of last item in the buffer

ICT1 - ICT4 - perform housekeeping for the input channel tapes; each table contains seven words whose functions are:

<u>WORD</u>	<u>BIT</u>	<u>FUNCTION</u>
0	0	channel busy bit indicator
0	32-49	number of input files not on this channel (OTHER)
1	0-63	current control word (CW) performing I/O; bit 27 used to indicate a read or write
2		spare
3	0-17	points to channel's I/O queue
	28-45	number of entries in the queue (QP)
4	0-23	bit address of Last File A-D
5	0-23	item - used for preselection
6	0-23	(LSTA - LSTD)
7	0-23	

The rest of the tables are used to accumulate statistics of certain timing procedures.

QP1 - QP5 - pointers to buffer queues

NEIS - Input buffer table (see EIS)

NEOS - Output buffer table (see EOS)

SCHR - bits 0-23 - start relative bit for duplicate comparison
 bits 32-55 - length to compare (64 reg.)

NCHR - 64 words used as intermediate storage

IEXP - 6 spare words

SKEY - bits 0-23 relative bit position of start key
 bits 32-49 length of key in bits - 64 words

Words In Common Storage

<u>WORD NAME</u>	<u>FUNCTION</u>	<u>BIT POSITION</u>
EIS	Input buffer stack pointer (NEIS) starting address of NEIS table number of entries in NEIS	0-17 28-45
EOS	Output buffer stack pointer (NEOS) starting address of NEOS table number of entries in NEOS	0-17 28-45
L	Input buffer length	0-17
M	Output buffer length	0-17
IL	Input item length (-sign)	0-23
ILLA	Input item length (+sign)	0-23
OUTIND	Output Edit Indicator	0
JSKPLB	First record skip indicator	0
NITEDT	Input Edit Indicator	0
JPTPLB	Label for output tape indicator	0
LABEL	The BCD label	0-63
HIKEY	A special stopper key of all 1's	12-59
NWDSK	Number of key words for the out- put edit to discard	32-49

Words In Common Storage (Cont'd)

<u>WORD NAME</u>	<u>FUNCTION</u>	<u>BIT POSITION</u>
MNOREC	Minimum number of bits for output items	32-55
MXOREC	Maximum number of bits for output items	32-55
IOVC	Item type	32-49
IOVBS	Output item byte size	32-49
OVNB	Number of length control bytes number of bytes in fill configuration	Ø-17 32-49
OVTL	Length bytes contents are binary or decimal indicator first length bytes' relative	Ø-17 32-55
OVTB	Termination bytes (right justified)	Ø-63
OVFL	Fill bytes (right justified)	Ø-63
BUFED	Starting address of last item processed	Ø-17
NCOMP	Number of sort keys for elim- ination of duplicate items	32-49
LMA	Bit size of output record	Ø-23
BUFIN	Address of temporary storage to be used at EOP time for pro- cessing input records	Ø-17
MINREC	Minimum input record size	Ø-23
MAXREC	Maximum input record size	Ø-23
INBS	Input item byte size	Ø-17
INVC	Item type	Ø-17

<u>WORD NAME</u>	<u>FUNCTION</u>	<u>BIT POSITION</u>
IVNB	Number of bytes in length control (VT or VL)	0-17
INCT	Mode of data binary or decimal origin of length control (VL)	0-17 32-55
IVTB	Terminator byte right justified (VT)	0-63
ISTD	Merging forward or backward	0
ICOLL	Collate indicator	0
NKEY	Number of input sort keys	0-17
IOVFC	Fill to maximum length indicator	32-49

SECTION V

CONTROL DECK ARRANGEMENT

There are two sources of merge parameters in a MERGE Job Deck:

1. The FIOD deck - The MCP REEL cards behind each input tape IOD card. For each physical input unit these REEL cards specify the number of reels to be merged, the reel labels, and the loading order of the reel. The MERGEP1 card relates each file to the proper physical unit(s). The job is usually run as a COMPILGO since the input arrangement may be varied.
2. Two or more parameter cards that follow the "SUBTYPE DATA" card in the System Card Reader. Each parameter card contains either MERGEP1, MERGEP2, or MERGEP4 in columns 1-7. Columns 73-80 of each card are ignored by the merge program and may be used in the usual way for external identification. Columns 9-72 contain, from left to right, 1-16 parameter fields, each 0-8 characters long, separated by commas. For each field, leading and trailing blanks are ignored. Null and blank fields are equivalent.

MERGEP1 Card

This card defines essential input parameters, indicated in the following list.

<u>FIELD #</u>	<u>CONTENTS</u>	<u>LEGAL VALUES</u>	<u>VALUE ASSUMED IF NULL</u>
1	Maximum input record length (bytes)	Any item, n, such that n*byte size <u>>192</u> bits	May not be null. Not applicable to FORTRAN.
2	Input data byte size	1 to 64 W = 64 B = 1 C8 = 8 C,C6 = 6	6
3	Should the merge skip the 1st record on each input tape?	Blank = No Nonblank = Yes	No.
4	Merge item length in words.	Integer	Meaningless if an input edit is specified. In that case this value is computed.
5	Is an input Edit required?	Blank = No Nonblank = Yes	No.
6	Number of logical inputs (M) to be merged.	$2 \leq M \leq 4$	May not be null.
7, 9, etc	For each logical input, the IOD (reference number can be obtained by examining position in FIOD deck) of tape drive on which odd-numbered reels of this input (1st, 3rd, etc.) are to be mounted.	Integer	Not allowed.
8,10,12, etc.	IOD of the tape drive on which even numbered reels (2nd, 4th etc.) of this input are mounted.	Integer	Not allowed.

MERGE2 Cards

These cards have two different formats which define parameters for the standard edit routine - they are not needed if input editing is not desired. These cards are the same as the SORTP2 parameters.

<u>F,FBIN</u>	<u>FIELD #</u> <u>VFBIN</u>	<u>VL,VT</u>	<u>CONTENTS</u>	<u>LEGAL VALUE</u>	<u>VALUE ASSUMED</u> <u>IF NULL</u>
1	1	1	Desired output Order	A = ascending D = descending	Ascending
2	2	2	Should key bytes be converted by table lookup before sorting:	S = Yes N = No	No conversion
3	3	3	Type of item	F = fixed length VL = variable length, specified in a field of each item. VT = variable length, marked by termination bytes. FBIN = fixed FORTRAN B or U records VFBIN = variable length B records	Fixed length
4	4	4	Fixed or minimum input items length (bytes)	1 to maximum input record length	6 words
	5	5	Maximum input item length (bytes)	1 to maximum input record length	6 words
			Type VL		
		6	Mode of field containing length	B = binary integer D = decimal 1 digit/byte	Binary

<u>F,FBIN</u>	<u>FIELD #</u> <u>VFBIN</u>	<u>VL,VT</u>	<u>CONTENTS</u>	<u>LEGAL VALUE</u>	<u>VALUE ASSUMED</u> <u>IF NULL</u>
		7	Relative origin of 1st length - containing byte (in bytes)	1 to minimum item length	Not applicable
		8	Length in bytes of field	1 to minimum item length	Not applicable
		Type VT			
		6	Entry mode of terminator configuration	A = A8 or A6 O = Octal; last three bits of each input byte.	A
		7	Terminator configuration	Any value legal for mode	Not applicable
		8	Length in bytes of terminator	1 to maximum item byte	Not applicable
5	6	9	Number of key fields	1 to 64	1

Remaining MERGEP2 Cards

<u>FIELD #</u>	<u>CONTENTS</u>	<u>LEGAL VALUE</u>	<u>VALUE ASSUMED</u> <u>IF NULL</u>
1	Relative origin of 1st key field (byte)	1 to minimum item length	Not applicable
2	Length of 1st key field (byte)	1 to minimum item length	"
3-4	Second key specification particulars	"	"
etc.	etc.	etc.	"

Additional keys may be input on MERGEP2 Cards as necessary to contain pairs of key specifications parameters.

MERGEP3 Cards

Essential output parameters are defined by MERGE3 cards.

<u>FIELD #</u>	<u>CONTENTS</u>	<u>LEGAL VALUE</u>	<u>VALUE ASSUMED IF NULL</u>
1	Maximum output record length	Any item, n, such that n*byte size \geq 192 bits.	May not be null Not applicable to FORTRAN
2	Output data byte	1 to 64 W = 64 B = 1 C8 = 8 C6,C = 6	6
3	Should the merge create a dummy label record on each output tape?	Blank = No Nonblank = Yes (up to 8 characters)	No
4	Is an output edit required?	Blank = No Nonblank = Yes	No
5	IOD of the tape drive on which the odd-numbered output reels (1st 3rd, etc.) are to be written.	Integer	Not allowed.
6	IOD of the tape drive on which the even numbered output reels (2nd, 4th etc.) should be written if different from 5 above.	Integer	Not allowed.

MERGEP4 Cards

This MERGEP4 card group defines parameters for the standard output edit routine; it is not needed if output editing is not desired.

<u>FIELD #</u>			<u>CONTENTS</u>	<u>LEGAL VALUES</u>	<u>VALUE ASSUMED IF NULL</u>
<u>F,FBIN</u>	<u>VFBIN</u>	<u>VL,VT</u>			
1	1	1	Type of item	F=fixed length VL=variable length specified in a field of each item. VT=variable length marked by terminator byte(s) FBIN=fixed FORTRAN B or U records VFBIN=variable length B records	Fixed length
2	2	2	Key length	Integers: ignored if input edit is specified.	Not applicable
3	3	3	Fixed or minimum	1 to maximum output record length	6 words
	4	4	Maximum output item length (bytes).	1 to maximum out- put record length	6 words
		Type VL			
		5	Mode of field containing length	B = binary integer	Binary
		6	Relative origin of 1st length containing byte	1 to minimum item length	Not applicable
		7	Length of field containing length (in bytes)	1 to minimum item length	Not applicable
		Type VT			
		5	Entry mode of terminator configuration	A=A6, A8 O=Octal	A6 or A8

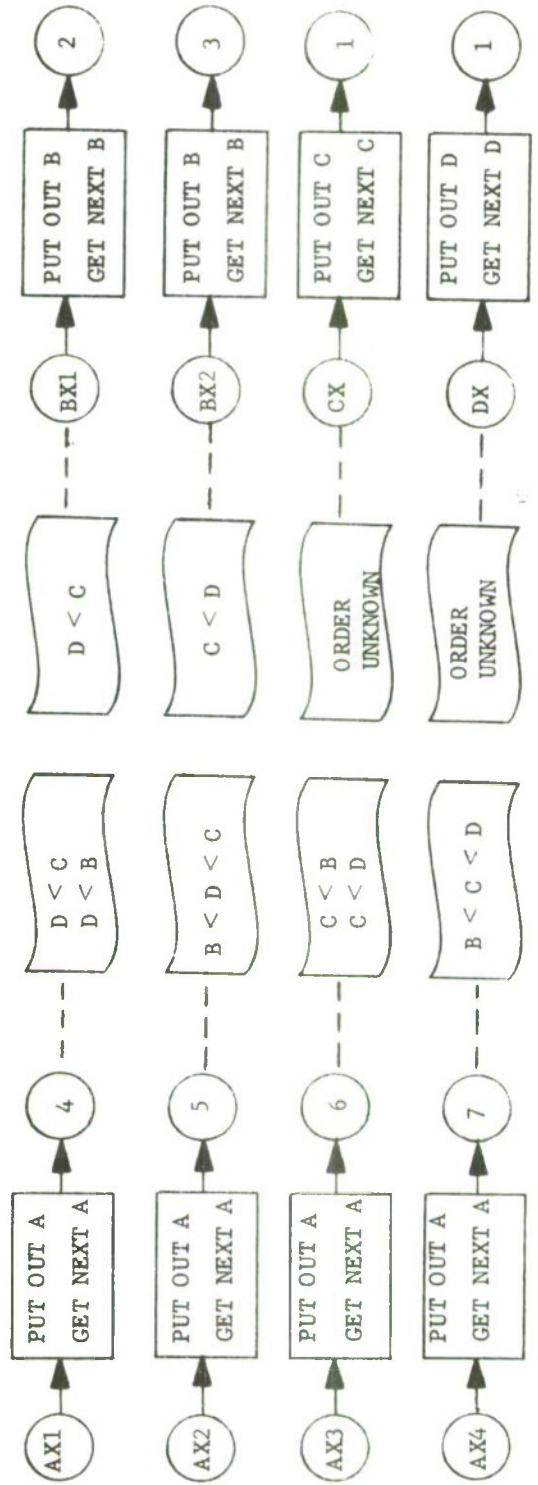
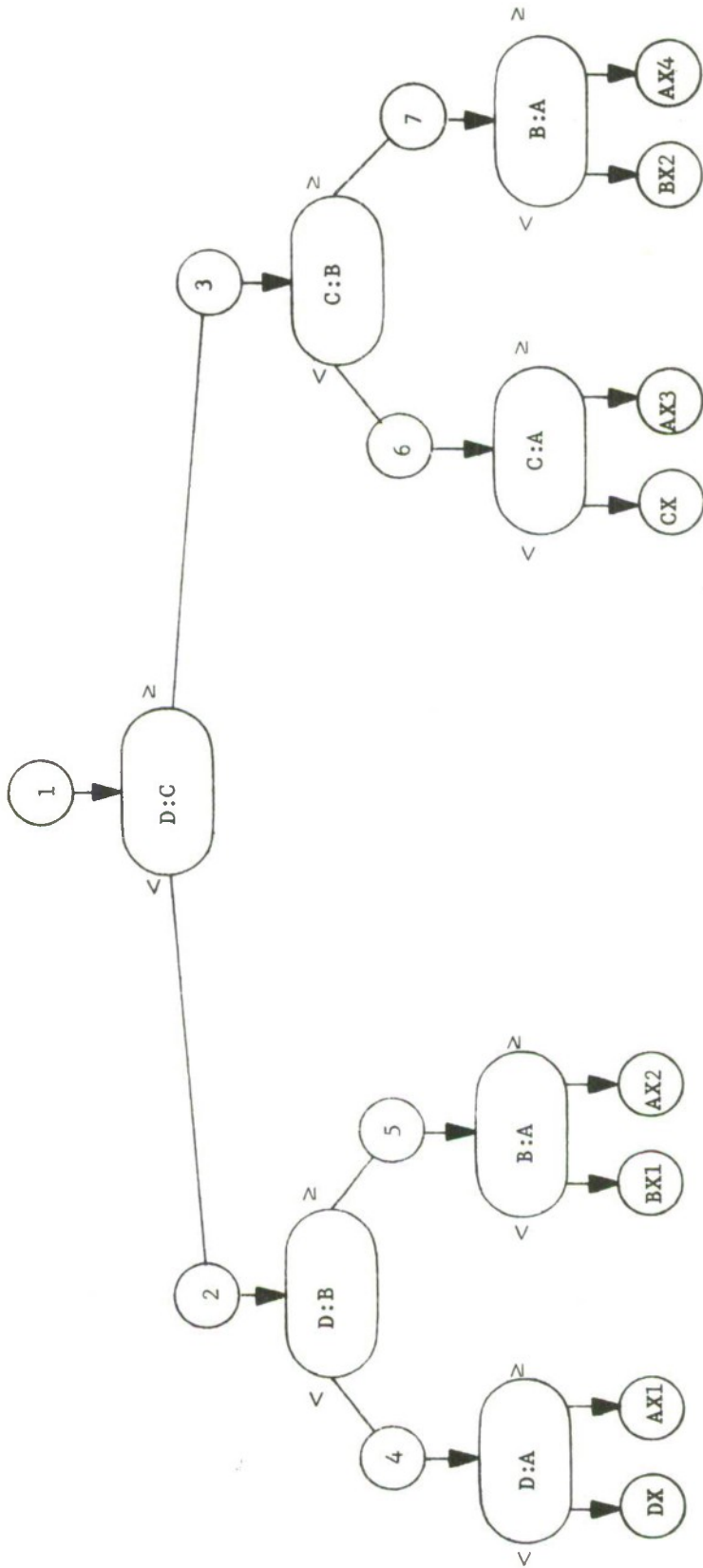
<u>F,FBIN</u>	<u>FIELD #</u> <u>VFBIN</u>	<u>VL,VT</u>	<u>CONTENTS</u>	<u>LEGAL VALUES</u>	<u>VALUE ASSUMED</u> <u>IF NULL</u>
		6	Terminator configuration	Any value legal for mode	Not applicable
		7	Length of bytes of terminator	1 to maximum item length	Not applicable
		Both VT and VL			
5		8	Mode of fill configuration of item is to maximum length.	A=A6, A8 O=Octal	No fill
6		9	Fill configuration	Any value legal for mode.	Not applicable
7		10	Length in bytes of fill configuration	1 to maximum size	Not applicable do not check for duplicate removal.
4	8	11	Number of fields to be compared for duplicate elimination	1-64	"

Second Type MERGEP4 Card

1			Relative origin of first key field (1 byte)	1 to minimum item length	Not applicable
2			Length of 1st key field (bytes)	1 to minimum length origin	Not applicable
3,4			Second key specifications parameter	"	"
etc.			etc.	etc.	etc.

Subsequent MERGEP4 cards may be used as necessary to contain pairs of key specifications that will not fit on the current card. The last card of the deck is "END" in columns 1-3.

APPENDIX



FOUR-WAY COMPARISON AND RETURN TREE

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) The MITRE Corporation Bedford, Massachusetts		2a. REPORT SECURITY CLASSIFICATION Unclassified	
		2b. GROUP	
3. REPORT TITLE MERGE			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) N/A			
5. AUTHOR(S) (First name, middle initial, last name) Beilin, Isaiah			
6. REPORT DATE August 1967		7a. TOTAL NO. OF PAGES 36	7b. NO. OF REFS 0
8a. CONTRACT OR GRANT NO. AF 19(628)-5165		9a. ORIGINATOR'S REPORT NUMBER(S) ESD-TR-67-371	
b. PROJECT NO. 503F		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) MTR-271	
c.			
d.			
10. DISTRIBUTION STATEMENT This document has been approved for public release and sale; its distribution is unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Deputy for Command Systems, Computer and Display Division; Electronic Systems Division, L. G. Hanscom Field, Bedford, Massachusetts.	
13. ABSTRACT A general purpose 7030 computer program has been written in the FORTRAN environment to merge in a single run of the program two, three, or four tape files which are the resulting output from a sort program. These files may be more than one tape in length and may be either FORTRAN or non-FORTRAN formatted. Input and output processors have been designed to handle various types of item formats; the program accepts items of fixed length, variable length - identified by a pre-designed terminator - and variable length with a length defining field. Upon request, the output processor will eliminate duplicate items and/or fill in variable length records to a standard size.			

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
COMPUTERS PROGRAMS MERGE STORAGE Program to merge data						